

University Politehnica of Bucharest
Faculty of Automatic Control and Computers
Computer Science and Engineering Department

Diploma Thesis

Mobile Gateway for Wireless Sensor Networks utilizing drones

by

Ioan Deaconu

Supervisor: Prof. Dr. Ing. Nicolae Tăpuş
Supervisor: As. Drd. Ing. Andrei Voinescu

Bucharest, September 2014

Contents

Contents	i
1 Introduction	3
2 Related Work	4
2.1 Standard WSN Protocols	4
2.2 Crop Monitoring	4
2.3 Aware platform	5
3 Hardware Platform	6
3.1 The Parrot AR.Drone 2.0	6
3.2 The Sparrow Family	7
3.2.1 The Sparrow Dongle	8
3.2.2 The SparrowV3.2	9
4 Software Implementation	10
4.1 The Debug Module	11
4.2 The Data Collecting Module	11
4.2.1 Modules intercommunication	12
4.2.2 Failure proof	12
4.3 The Communication Module	12
4.3.1 Socket with connection reset	13
4.3.2 JSON Encoding of Data	13
4.4 Android application modules	13
4.4.1 Display information module	14
4.4.2 FTP communication module	15
5 Testing	16
5.1 Scenario	16
5.2 Results	16

6 Future Work	17
6.1 Features	17
6.1.1 Debug	17
6.1.2 Truely Autonomous flight	17
6.2 Applications	17
6.2.1 Treasure Hunt Games	17
6.2.2 Search and Rescue	18
7 Conclusions	19
7.1 Outlook	19
Bibliography	20
List of Figures	21
List of Tables	22
Listings	23

Abstract

Keywords Wireless Sensor Networks, task, scheduling, graph cuts

Acknowledgements

Chapter 1

Introduction

Thesis Intro - no more than 3 pages.

Chapter 2

Related Work

Research has been done in the area of wireless sensor networks with mobile gateways, but very few include the same constraints as in our networks. Such constraints consist of long distances between nodes, simple, fast implementation and the possibility to run

Where most research focuses on routing packets to a mobile sink, we consider applications where network lifetime and low-maintenance are more important, and as such energy consumption on the nodes must be kept to a minimum (in order to have 3+ years of network lifetime with no maintenance/battery changes). Shifting the energy consumption to the sink is a direct consequence of this decision, under the assumption that there exists a flight path that allows the gateway to collect information from all the nodes.

2.1 Standard WSN Protocols

The protocols implemented in Wireless Sensor Network are based on surrounding node discovery in order to build a topology and find the best way they can multihop data to the gateway. This approach works best in a static environment, but in a dynamic environment or an environment where the distance between nodes is too big or the time between two data packets is too big, the network convergence will be slow or not even possible.

2.2 Crop Monitoring

A research of using a drone for crop monitoring has been conducted at a vineyard. Their system was comprised of an unnamed quadcopter, an arduino with a gprs module for theoretical long distance communication with drone, but the

drone was controlled using a Spektrum DX7SE 2.4 GHz remote control and ZigBee and Crossbows TelosB as the nodes.

They demonstrated that a preprogramed UAV can be used to monitor multiple crops were a standard WSN cannot be applied because of the unique constrains imposed by the the environment.

The cost of the implementation was relatively high compared to ours, the remote is 300\$, the same as the entire drone that we propose and the TelosB is 99\$. This data suggests that for their experiment the drone, communication module and the remote control were half the cost of the equipment.

Another problem was that they were not saving the data localy, but sending it back to the base station were it was proccesed and saved. This cand represent a problem because the system cannot function propely unless a base station is supplied.

2.3 Aware platform

The Aware platform, proposed by Ays. Egl Tysz Erman, Lodewijk Van Hoesel and Paul Havinga from University of Twente, is a platform that integrates WSNs, UAVs, and actuators into a disaster response setting and provides facilities for event detection, autonomous network repair by UAVs, and quick response by integrated operational forces[EHHW08].

They uses the UAVs to deploy new nodes that will replace the damaged ones an check if they function. The entire system still relies on a sink to collect the data and to send them to a base station.

Chapter 3

Hardware Platform

In this chapter we will present the hardware platforms used.

Because we wanted to emphasise not only a new way of aquiring data, but also a simple and low cost one, we have selected The Parrot AR.Drone 2.0 as the work horse that will carry the gateway that will communicate with the nodes from our Sparrow Family .

chesti help .. anybody

3.1 The Parrot AR.Drone 2.0

Parrot AR.Drone is a wifi radio controlled flying quadcopter built by the French company Parrot. The original drone was released in 2010 and in 2012 it was replaced by version 2.0. Since the launch of the original AR.Drone, more than half a million units have been sold, making it one of the, if not, the most popular drone on the market.

The reason of its success is not entirely due to the relatively low price of around 300\$ but also because it is very easy to learn how to control the drone and also because of the usb port that accomodate any device using that interface and the linux operating system

Because of those reasons, the Drone has a number of aftermaket modules that can be attached to it like the Flight Recorder GPS Module. This module has a built in storage of 4GB for video recording purposes and a built in GPS receiver. This allows the drone to follow a predetermined path of waypoints and to return back from where it took off automatically, all within the limit of the Wi-Fi connection with the control device.

The arrot AR.Drone 2.0 specifications are :

- 1GHz 32 bit ARM Cortex A8 processor with 800MHz video DSP TMS320DMC64x



Figure 3.1: *The arrot AR.Drone 2.0*

- Linux 2.6.32
- 1Gbit DDR2 RAM at 200MHz
- USB 2.0 high speed for extensions
- Wi-Fi b,g,n
- 3 axis gyroscope 2000/second precision
- 3 axis accelerometer +-50mg precision
- 3 axis magnetometer 6 precision
- Pressure sensor +/- 10 Pa precision
- Ultrasound sensors for ground altitude measurement
- 60 fps vertical QVGA camera for ground speed measurement
- 30 fps 720p front mounted camera

3.2 The Sparrow Family

The Sparrow Family , composed of the Sparrow Dongle and SparrowV3.2, use a 2,4 GHz wireless network as a medium of communication.

At heart,soul and brain of this family lies the ATMega128RFA1. It is an 8bit microcontroller from Atmel that has a very important on-chip 2.4 GHz wireless transceiver that helps in keeping the footprint of the board very small.

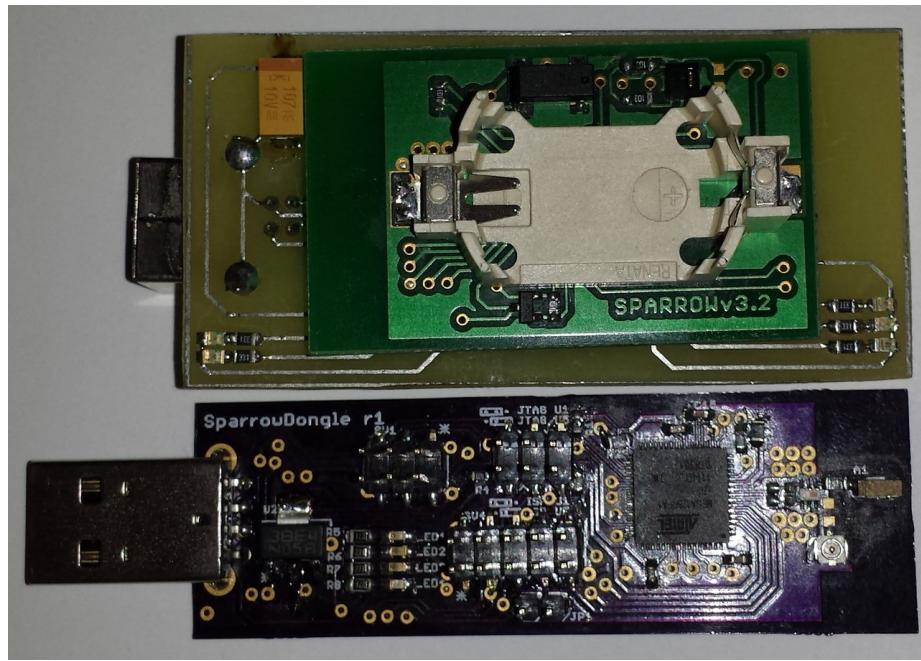


Figure 3.2: The Sparrow Dongle next to the SparrowV3.2

The signal received or sent by the wireless transceiver can be boosted by attaching an external antenna. For example, in an ideal situation, with no interferences from the outside world , an 8-dbi omni-directional antenna mounted on both communication devices would amount to an around 200 meters of communication range, well over the 70 meters measured with the default antennas.

3.2.1 The Sparrow Dongle

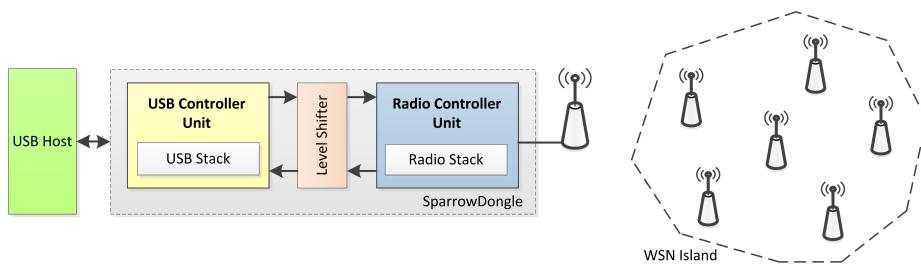


Figure 3.3: SparrowDongle stick architecture

The unseen hero of the WSN world, the Sparrow Dongle is the gateway of the Sparrow Family. The Dongle can be connected to any devices that has an

usb port and can support USB CDC with ACM module (USB Communications Device Class with Abstract Control Mode).

The dongle uses Atmega32U4 as a dedicated USB Controller unit. This design allows the RF controller to run any RF communication stack without having the USB code intrude on key timings.

3.2.2 The SparrowV3.2

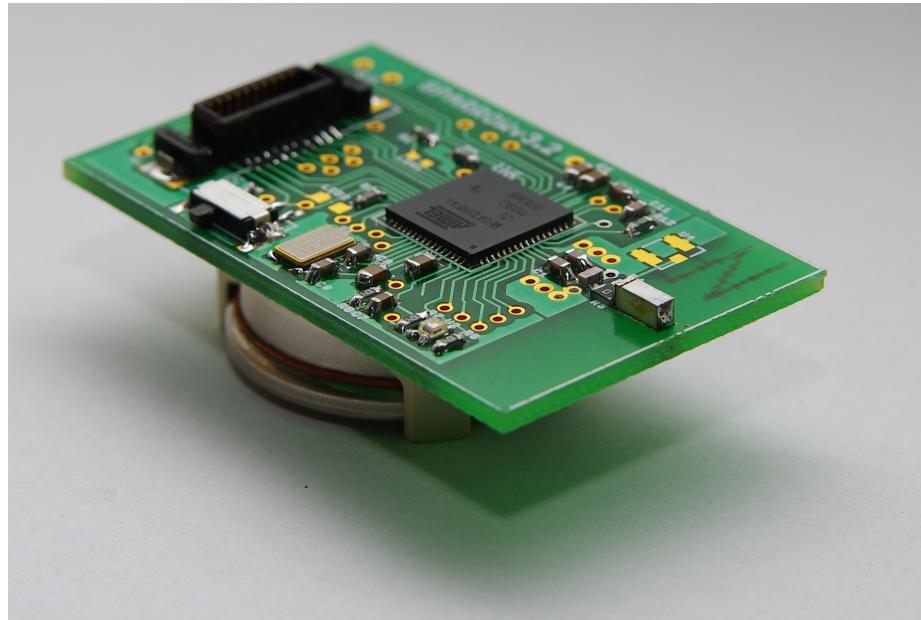


Figure 3.4: *The SparrowV3.2*

The main hero of The Sparrow Family, the SparrowV32 nodes does all the dirty work of collecting data from the surrounding environment and send it to the Sparrow Dongle. The collected data depends on the sensors of the SparrowV3.2. The standard implementation has a light, temperature and humidity sensor. Besides this information, it also sends the battery level.

Chapter 4

Software Implementation

The entire solution is divided into different modules that run independently but communicate with each other to achieve the main purpose. The idea of software organized as modules allows for future features to be added more easily.

The main modules are installed in the AR Parrot Drone and the android FreeFlight 2.0 application.

The SparrowDongle gateway is always in a listen for data state and dumps every data received on the serial. When it receives the data, it sends back an ack to let the SparrowV3.2 node to know that it can begin sending the entire stored data to the mobile gateway.

The SparrowV3.2 node is sending periodically a small data packet to check if a gateway is available. When it receives the ack for the packet it starts sending the stored data to the gateway. The data sent can vary, from sensor readings to debugging informations in order to check the state of the Wireless Sensor Network.

The data gathered by the gateway is saved into different files in the AR Parrot Drone's internal memory. The files also contains informations such as the node identification tag and time of the transfer. The data can be accessed at any time by any device connected to the drone's wireless network port 4242 via FTP.

The drone also processes some collected data to provide informations like signal strength, last connection time and number of nodes. This informations are send to the controlling device through a socket connection.

The controlling device, pc or android, will gather the informations and display them to the user.

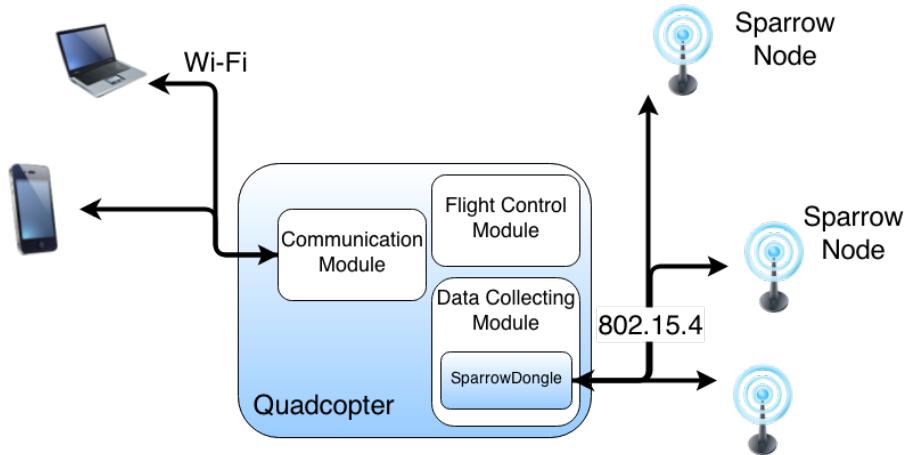


Figure 4.1: *Modules and connections between them and devices*

4.1 The Debug Module

The code can be compiled to display debug informations to the console or to suppress them entirely. Also, in certain parts of the modules, a wait action is need in order to wait for an action to be executed. Now the delay is set at 100 ms, but it can be easily modified to any desired value.

Listing 4.1: Data Collection use of mutex

```
/* activates/deactivates printf debuf information*/
#define DEBUG_ON 0
/* delay yime in microseconds*/
#define DELAY_US 100000
#define DEBUG_PRINT(a...) { if(DEBUG_ON) printf(a); }
```

4.2 The Data Collecting Module

The module saves the collected data into the drone's internal memory and passes the data in order to gather certain informations like number of nodes currently connected to the Dongle, signal strength, if the Dongle is connected etc. This informations are passed to the communication module to provide to the user realtime feedback.

This module, besides the main purpose and similar to the other modules, has some extra features that are designed to make the solution more user friendly and easier to improve in the future.

4.2.1 Modules intercommunication

The memory area in which the informations sent to the user are saved is shared between this module and the communication module. Basically, the way this two modules interact with each other can be compared to the consumer - producer problem, where the Data Collecting Module can be associated with the producer side and the Communication Module with the consumer side.

The main problem consists of deadlocks and data starvation. This is prevent with the use of one mutex that allows only one thread at a time to modify the informations.

Listing 4.2: Data Collection use of mutex

```
pthread_mutex_lock(&data_lock);
add_node_data(get_current_timestamp(), read_data + 7);
pthread_mutex_unlock(&data_lock);
```

The mutex is used similarly in the Communication Module when it consumes the information.

4.2.2 Failure proof

Because the Dongle is connected to an USB port on a machine that has a lot of vibrations, it might disconnect / reconnect for a very short period of time, so this module has been designed with multiple USB disconnects and reconnects without the need to reset the Drone. This information is vital, because you can check if the Dongle is still connected to the dron without the need to inspect it visually or to connect to a debug terminal.

4.3 The Communication Module

All the information gathered by the Data Collecting Module would be useless if it cannot be accessed easily.

This module, as the name suggests, handles the communication of this crucial information back to the user.

Being a different module, with different attributions than the Data Collecting Module, it has an entire process dedicated to it for 3 important reasons:

- 1. This approach of a module with its own process allows the modules to run independently of each other;
- 2. The Data Collecting Module can collect the data from the Dongle as soon as it has a new one available;
- 3. If the Communication Module stops working, the Data Collecting Module continues to save the new informations received.

4.3.1 Socket with connection reset

The communication is done through socket connections listening on port 8888. It accepts only one connection at a time.

If a connected client decides to disconnect before or while a write is performed, a SIGPIPE error signal will be thrown, stopping all the modules. This is prevent by ignoring the signal, forcing the write action to return a EPIPE, an exiting the this function.

The main process will callback the accept_socket_connection to reestablish a new connection. Once a connection is established, it will send information once every DELAY_US microseconds. The program was configured and tested with a 100 ms wait period that leads to a ten times per second information update.

This delay is required because:

- If data is sent too often, the socket might be flooded and stop sending the data
- If there were no delay, it will create a big and useless processor busy state both for the drone and the controlling device.

4.3.2 JSON Encoding of Data

The data sent is encoded in JSON format, because it is very easy to encode and all devices can decode it.

The informations encoded are

- Dongle connection status
- An array containing node informations
 - Node unique ID
 - Last connection time of the node to Dongle
 - The power of the received signal

4.4 Android application modules

Being an opensource platform we have modified the AR Freeflight 2.0 android application to communicate with our new modules added to the drone.

Android fairly imposes the use of the background process class AsyncTask when you have to use communication protocols like http, ftp, sockets. This prevents the ui process from being stuck in communication and not responding to user actions.

The class offers 5 very important methods that can be overwritten, 3 running on the main ui process, that prepare data before and after execution, publish the

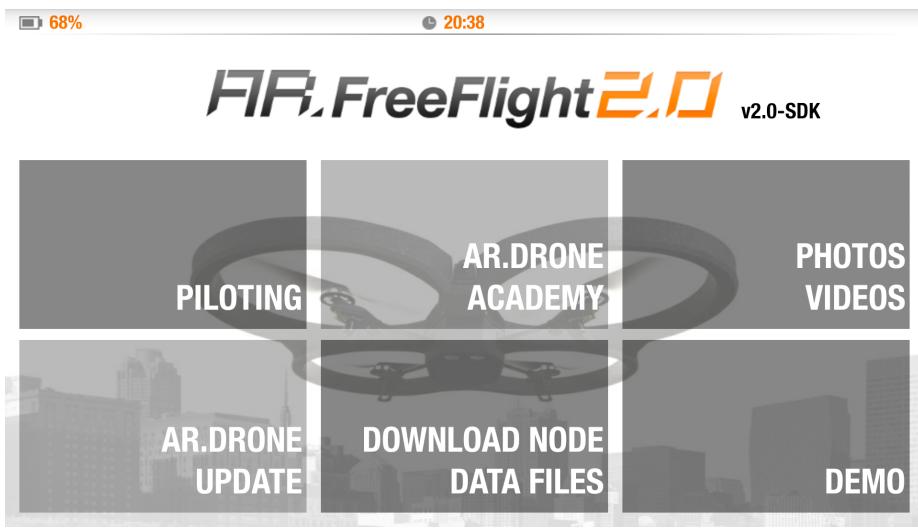


Figure 4.2: *ARFreeflight modified application*

progress or simply cancel at any step, and 1 running on the actual background process.

4.4.1 Display information module



Figure 4.3: *ARFreeflight modified Piloting Screen*

The Piloting screen of the application has been modified to display the received informations from the drone.

The information displayed consists of the dongle being functional and at most, 9 nodes sorted descending after their signal strength.

4.4.2 FTP communication module

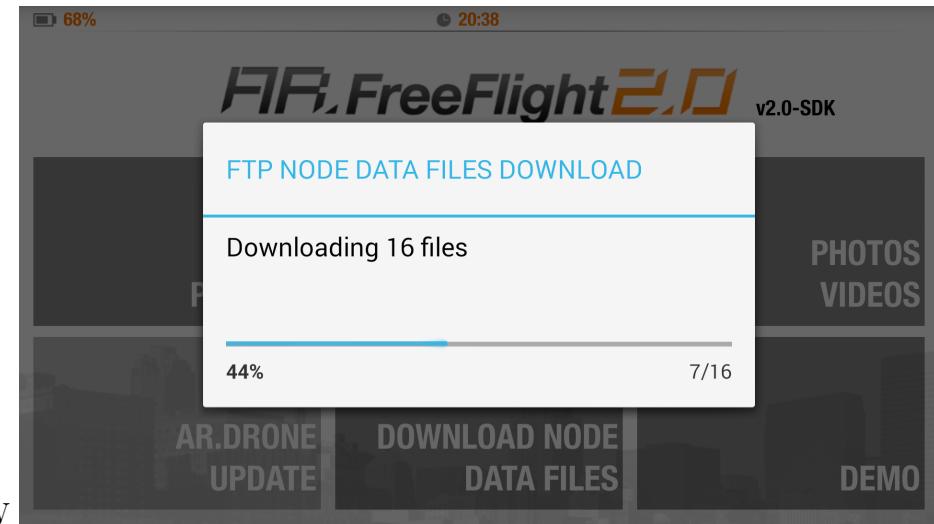


Figure 4.4: *ARFreeflight* FTP downloading files

The drone has a build in FTP server that can be configured to allow acces to any folders/files on the drone. We have configured the drone so that the folder that contains the saved data can be accesed at any time using the 4242 port by any device that has FTP client capabilities. We have added this feature to the android application as well.

The application will download all the files from the drone to the local storage of your android device while displaying the progress.

Chapter 5

Testing

5.1 Scenario

5.2 Results

Chapter 6

Future Work

Even though the solution is working, new application and functionalities can be added. The application can be from opposing worlds, such as search and rescue versus military.

6.1 Features

6.1.1 Debug

A feature that can be used in conventional wireless sensor networks is to determine the source of communication failure. If the gateway detects that the network has a communication problem and not all de previous nodes can be reached, this information will be send to the drone and it will try to find and determin which nodes are working properly and which nodes are not.

6.1.2 Truely Autonomous flight

The AR Parrot Drone 2.0 can perform autonomous flight with a gps module, but only while it is still in the range of the wifi connection.

6.2 Applications

6.2.1 Treasure Hunt Games

A treasure hunt game can be played by placing the nodes with information regarding the location of the treasure at certain hidden spots and trying to find the treasure by following the clues provided by the drone.

6.2.2 Search and Rescue

When tourists go hiking or other similar activities and get lost they can be found if they will wear a wireless sensor that monitor vitals and can gather a periodical gps position. The drone will search for the signal emitted by the node and help pinpointing their exact location while offering a live video feed of them.

Chapter 7

Conclusions

7.1 Outlook

Bibliography

- [EHHW08] Aysegül Tüysüz Erman, LV Hoesel, Paul Havinga, and Jian Wu. Enabling mobility in heterogeneous wireless sensor networks cooperating with uavs for mission-critical management. *Wireless Communications, IEEE*, 15(6):38–46, 2008. [cited at p. 5]

List of Figures

3.1	<i>The arrot AR.Drone 2.0</i>	7
3.2	<i>The Sparrow Dongle next to the SparrowV3.2</i>	8
3.3	<i>SparrowDongle stick architecture</i>	8
3.4	<i>The SparrowV3.2</i>	9
4.1	<i>Modules and connections between them and devices</i>	11
4.2	<i>ARFreeflight modified application</i>	14
4.3	<i>ARFreeflight modified Piloting Screen</i>	14
4.4	<i>ARFreeflight FTP downloading files</i>	15

List of Tables

Listings

4.1	Data Collection use of mutex	11
4.2	Data Collection use of mutex	12